1. A method of graphically rendering a virtual object, the method comprising the steps of:

(a) using an index corresponding to each of a plurality of jacks of a voxel-based virtual object to identify texture elements for which surface elements of the virtual object are mapped; and

(b) generating texture coordinates of the identified texture elements in a texture space.

2. The method of claim 1, wherein the index corresponds to one of a known plurality of surface element configurations produced during mesh generation.

3. The method of claim 1, wherein the index is a marching cubes index or a marching tetrahedra index.

4. The method of claim 1, wherein the texture space comprises a plurality of texture regions.

5. The method of claim 4, further comprising the steps of:

(c) binding to a graphics application a blended texture corresponding to one of the plurality of texture regions; and

(d) transmitting to the graphics application the texture coordinates of the texture elements of the blended texture.

6. The method of claim 1, further comprising the step of:

(c) dividing an object space comprising at least a portion of the virtual object into a plurality of jack blocks each comprising a plurality of jacks, wherein step (b) comprises generating texture coordinates of texture elements to which surface elements within each jack of each jack block are mapped.

7. The method of claim 1, wherein step (a) comprises using a first lookup table to determine to which of a plurality of texture elements a surface element is mapped.

8. The method of claim 7, wherein step (a) comprises using a second lookup table to determine to which of a plurality of positions within a texture element a surface element is mapped.

9. The method of claim 7, further comprising the step of creating the first lookup table.

10. The method of claim 1, wherein the texture elements are quadrilaterals and the surface elements are triangles.

11. The method of claim 10, wherein step (a) comprises assigning either one or two triangular surface elements to a quadrilateral texture element.

12. The method of claim 11, wherein step (a) comprises assigning to a quadrilateral texture element two triangular surface elements that share a common edge.

13. The method of claim 1, wherein the surface elements of the virtual object are components of a triangle mesh generated for the virtual object using at least one of a marching cubes algorithm and a marching tetrahedra algorithm.

14. A method of mapping texture onto a virtual object, the method comprising the steps of:

(a) at least one of:

(i) allocating texture in a texture space for at least one newly-created jack of a virtual object following an object modification; and

(ii) de-allocating texture in the texture space for at least one newly-eliminated jack of the virtual object following an object modification; and

(b) rendering the virtual object.

15. The method of claim 14, wherein the texture is spatially incoherent.

16. The method of claim 14, wherein step (a) comprises determining, in a stepwise manner for each of a plurality of jacks, whether texture elements have previously been assigned to surface elements in that jack.

17. The method of claim 16, wherein step (a) proceeds from a first jack to a second jack in render order.

18. The method of claim 16, wherein step (a) comprises determining whether there are sufficient free texture elements in a current texture region in which to allocate texture for the jack.

19. The method of claim 18, wherein step (a) comprises creating a new texture region where there are insufficient free texture elements in the current texture region for the jack.

20. The method of claim 19, wherein step (a) comprises keeping within a single texture region all texture elements assigned to all jacks of a jack block.

21. The method of claim 14, wherein step (b) comprises generating, for surface elements of the virtual object, coordinates of corresponding texture elements in the texture space determined according to a marching cubes index for each of a plurality of jacks of the virtual object.

22. The method of claim 14, wherein step (a) and step (b) are each performed automatically, and wherein the method obviates creating a global parameterization.

23. The method of claim 14, wherein step (a) comprises retaining texture previously allocated for at least one jack that is unchanged following the object modification.

24. The method of claim 14, wherein step (a) comprises generating, for a given texture region, a hash map comprising an identification of each of at least one texture element in each of a plurality of jacks.

25. The method of claim 14, wherein step (a) comprises generating, for a given texture region, a hash map of hash maps comprising an identification of each of at least one texture element in each jack of at least one jack block.

3

26. A method of creating a blended texture for use in rendering a virtual object, the method comprising the step of blending a plurality of texture layers, at least one of which has associated therewith a grid indicating at least one member of each of at least two of:

(i) a uniform texture element;

(ii) a nonuniform texture element; and

(iii) a location of a nearest free texture element in the grid.

27. The method of claim 26, wherein the grid indicates at least one member of each of (i), (ii), and (iii).

28. The method of claim 26, wherein an element of the grid indicating a uniform texture element is a uniform value associated with the uniform texture element.

29. The method of claim 28, wherein the uniform value indicates at least one of:

(a) an intensity; and

(b) a color.

30. The method of claim 26, wherein an element of the grid comprises a pointer indicating a nonuniform texture element refers to a quadrilateral of texels.

31. The method of claim 30, wherein the quadrilateral comprises two triangles in texture space mapped to two contiguous surface elements of the object.

32. The method of claim 26, wherein the step of blending the plurality of texture layers comprises performing at least one compositing operation.

33. A method of creating a blended texture for use in rendering a virtual object, the method comprising the step of blending a plurality of texture layers comprising at least one member of each of at least two of:

(i) a scratch texture;
(ii) a stencil texture; and
(iii) a paint texture.

34. The method of claim 33, wherein at least one of the plurality of texture layers has associated therewith a grid of pointers indicating at least one member of each of at least two of:

(a) a uniform texture element;
(b) a nonuniform texture element; and
(c) a location of a nearest free texture element in the grid.

35. The method of claim 33, wherein the plurality of texture layers comprise at least one member of each of (i), (ii), and (iii).

36. The method of claim 33, wherein the step of blending the plurality of texture layers comprises performing at least one compositing operation.

37. The method of claim 33, wherein the step of blending a plurality of texture layers comprises blending a first set of at least two texture layers each corresponding to a first

texture region, and blending a second set of at least two texture layers each corresponding to a second texture region.

38. The method of claim 37, wherein the step of blending a plurality of texture layers comprises, for each of a plurality of texture regions, blending a set of at least two texture layers corresponding to a respective texture region.

39. The method of claim 33, wherein the plurality of texture layers comprises a scratch texture corresponding to a single brush stroke performed by a user.

40. The method of claim 39, wherein the single brush stroke is selected from the group consisting of a paint stroke, an erase stroke, a pencil stroke, a pen stroke, a line application, a character application, and a text application.

41. The method of claim 39, wherein the single brush stroke extends from a first texture region into a second texture region.

42. The method of claim 39, wherein the brush stroke is divided into a plurality of segments.

43. The method of claim 42, wherein an intensity value corresponding to a location within the brush stroke is written at a texel of the scratch texture only if the intensity value exceeds an existing value at the texel.

44. The method of claim 39, wherein a value written at a texel of the scratch texture is a function of a distance between a point in object space corresponding to the texel and a point in object space along a center of the brush stroke.

45. The method of claim 44, wherein the point in object space corresponding to the texel and the point in object space along the center of the brush stroke are represented with Cartesian coordinates.

46. The method of claim 33, wherein the step of blending comprises blending a scratch texture corresponding to a single brush stroke with at least a second texture substantially upon completion of the brush stroke.

47. The method of claim 33, wherein the plurality of texture layers comprises a plurality of scratch textures, each corresponding to a single brush stroke performed by a user.

48. The method of claim 47, wherein the step of blending comprises blending the plurality of scratch textures in an order in which the brush strokes corresponding to the scratch textures are performed.

49. The method of claim 33, wherein the step of blending proceeds while a user performs brush strokes.

50. The method of claim 49, wherein the step of blending proceeds in a plurality of discrete time slices.

51. The method of claim 33, wherein the plurality of texture layers corresponds to a first texture region, and wherein the method further comprises the steps of:

(i) creating a first stencil texture comprising a plurality of texels, each with an assigned scalar value indicating a level of protection for a corresponding texel in at least a portion of the first texture region;

(ii) directing graphical input corresponding to a plurality of brush strokes performed by a user into a second stencil texture; and

(iii) modifying a value of at least one texel of the second stencil texture using the first stencil texture.

52. A method of interactively representing application by a user of at least one brush stroke directly onto a virtual object in object space, the method comprising the steps of:

(a) allocating a plurality of texture elements in two dimensional texture space for a plurality of jacks of a virtual object;

(b) graphically rendering the allocated texture in real time as a user applies at least one brush stroke onto the virtual object as represented in object space, wherein the rendering step comprises creating at least one blended texture that is at least temporarily bound to a graphics application during the rendering; and

(c) updating at least one of the blended textures according to the at least one brush stroke applied by the user, wherein the method further comprises at least one of:

(i) using an index corresponding to each of a plurality of jacks of the virtual object to identify texture elements to which surface elements of the virtual object are mapped;

(ii) allocating texture in the texture space for at least one newly-created jack of the virtual object following an object modification; and

(iii) blending a set of texture layers corresponding to a first of a plurality of texture regions in the texture space and binding the blended texture to the graphics application during rendering of the first texture region.

53. The method of claim 52, wherein the method further comprises at least two of:

(i) using an index corresponding to each of a plurality of jacks of the virtual object to identify texture elements to which surface elements of the virtual object are mapped;

(ii) allocating texture in the texture space for at least one newly-created jack of the virtual object following an object modification; and

(iii) blending a set of texture layers corresponding to a first of a plurality of texture regions in the texture space and binding the blended texture to the graphics application during rendering of the first texture region.

54. The method of claim 52, wherein the method further comprises:

(i) using an index corresponding to each of a plurality of jacks of the virtual object to identify texture elements to which surface elements of the virtual object are mapped;

(ii) allocating texture in the texture space for at least one newly-created jack of the virtual object following an object modification; and

(iii) blending a set of texture layers corresponding to a first of a plurality of texture regions in the texture space and binding the blended texture to the graphics application during rendering of the first texture region.

55. The method of claim 52, wherein the at least one brush stroke comprises at least one member selected from the group consisting of a paint stroke, an erase stroke, a pencil stroke, a pen stroke, a line application, a character application, and a text application.

56. The method of claim 52, further comprising the step of:

(d) haptically rendering the virtual object in real time as the user applies the at least one brush stroke onto the virtual object.

57. The method of claim 56, further comprising the step of:

(e) determining force corresponding to a position of a haptic interface device held by the user as the user applies the at least one brush stroke onto the virtual object.

58. The method of claim 57, further comprising the step of:

(f) providing the force to the user through the haptic interface device as the user applies the at least one brush stroke onto the virtual object.

59. The method of claim 56, wherein step (d) is performed at a substantially faster rate than step (b).

60. The method of claim 56, wherein step (d) is performed within a range from about 700 Hz to about 1500 Hz and wherein step (b) is performed within a range from about 5 Hz to about 150 Hz.

61. The method of claim 60, wherein step (d) is performed at about 1000Hz and wherein step (b) is performed at up to about 40 Hz.

62. The method of claim 56, wherein step (b) and step (d) are performed by different threads.